

基于 DBSCAN 的簇共享对象的处理办法

徐仰彬, 刘志镜

(西安电子科技大学 计算机学院, 陕西 西安 710071)

摘要:聚类分析是数据挖掘的一类主要的方法,它可以自动根据相似性对数据对象进行分组,发现数据空间的分布特征。DBSCAN 算法是经典的基于密度的聚类算法,针对此算法处理簇边界共享点的不足之处,改进了此算法。试验结果证实了确实可以提高聚类结果的质量。

关键词:数据挖掘;聚类分析;DBSCAN 算法

中图分类号:TP301.6

文献标识码:A

文章编号:1673-629X(2007)07-0038-02

A DBSCAN - Based Algorithm for Boundary Object of Cluster

XU Yang-bin, LIU Zhi-jing

(Computer School, Xidian University, Xi'an 710071, China)

Abstract: Cluster analysis is a primary method for data mining. The DBSCAN algorithm can discover clusters of random shape from database. Improved the algorithm in dealing with boundary object of cluster. The results of the experiments demonstrate that this algorithm can improve the cluster quality surely.

Key words: data mining; cluster analysis; DBSCAN algorithm

0 引言

按密度划分的聚类方法 DBSCAN 算法拥有可以发现任意形状的簇和有效的屏蔽噪声数据干扰的优点,因此被广泛地应用于数据挖掘中。但是纵观近几年的关于聚类方面的研究更多的是集中到针对算法的效率提高上,对于聚类结果的准确性的论述相对较少,一定程度上忽略了聚类的质量。尤其是算法对共享点的处理方式不是很合理。针对此不足之处,我们做了大量的研究,通过吸取前人的宝贵经验,提出了采用距离密度函数和相对互连函数来改进算法,使对边界点的处理更加合理化,同时又能保持较高的执行效率。

1 DBSCAN 算法简述

1.1 算法基本思想

DBSCAN (Density - Based Spatial Clustering of Applications with Noise) 是一个基于密度的聚类算法。该算法将具有足够高密度的区域划分成簇,并可以在带有噪声的空间数据库中发现任意形状的聚类。它定义

簇为密度相连的点的最大集合。此算法使用两个参数: ϵ - 邻域和 MinPts, 分别代表了给定对象的区域半径和此区域中所包含的对象点的数目。

1.2 算法描述

DBSCAN 通过检查数据库中每个点的 ϵ - 邻域来寻找聚类。如果一个点 p 的 ϵ - 邻域包含多于 MinPts 个点,则创建一个以 p 作为核心对象的新簇。然后, DBSCAN 反复地寻找从这些核心对象直接密度可达的对象,这个过程涉及到一些密度可达簇的合并。当没有新的点可以被添加到任何簇的时候,该过程结束^[1]。

2 DBSCAN 算法的缺点和已有的改进方法

2.1 算法的缺点

当两个簇在边界处相交时,即存在这样的点,是非核心点但同时和两个簇中的核心点均密度相连。DBSCAN 算法简单地将其分到先处理的那一核心点所属的簇中。这样做存在很大的问题。其一: 这样处理使得结果对输入顺序敏感,这显然不是我们所想的。其二: 虽然从处理结果中来看,这是两个簇,但是被分开的子簇共享了其中的某些对象。当改变 ϵ - 邻域值的时候,很可能这两个簇就会合并成一个簇。由于 DBSCAN 对参数 ϵ 的敏感性,会导致聚类结果的巨大差异^[2,3]。

收稿日期: 2006-09-12

基金项目: 国家自然科学基金资助项目 2005(60573139)

作者简介: 徐仰彬(1979-), 男, 山东沂南人, 硕士研究生, 研究方向为数据挖掘; 刘志镜, 教授, 研究方向为数据挖掘、人工智能、计算机应用。

2.2 已有的改进算法和存在的不足之处

参考文献[4]中采用了记录簇连接信息的方式发现所有的相交簇。但是在后面的处理中采取了交互的方式。手工进行簇的合并。在一定程度上可以提高聚类的效果。但是在数据聚类上缺乏连贯性,而且当数据集很大的时候聚类结果的可视化会成为性能上的瓶颈[4]。

参考文献[5]采用基于方向的聚类方法来判断共享点的归属。但是缺少对 ϵ - 邻域半径的有效的修正,可能存在的错误将簇分成几个簇的问题没有提到[5]。

3 此改进算法的主要思想

本改进算法采用了核心距离密度函数来判断边界点的归属。采用相对互连性和相对近似来决定簇间的相似性,从而确定是否有必要合并两个相交的簇。这样就可以有效地解决以上两种算法存在的不足。

首先采用 DBSCAN 算法进行聚类,聚类的同时记录下来所有的簇间的共享点。由于共享点远远少于数据库中的记录点。所以增加的时间可以忽略不计。首先判断共享点所对应的几个簇是否具有较大的相似性。通过设定一个相似阈值来确定。如果超过这个值。则认为应该将两个簇合并,产生这种错误分类的原因是 ϵ - 邻域的选择很不合适。这样可以有效地减小对输入参数的敏感性。当小于这个值的时候,则认为不是不同的类,然后通过计算共享点到其核心点的距离核心密度来判断此点属于哪个簇更合适。

4 算法中用到的两个核心公式

判断两个拥有共享点的子簇是否应该合并的方法:采用相对互连性 $RI(C_i, C_j)$ 来判断。

$$RI(C_i, C_j) = \frac{R(C_i - C_j)}{\frac{1}{2} \left[\frac{\sum (C_i - X_j)}{N_j} + \frac{\sum (C_j - X_i)}{N_i} \right]}$$

其中: C_i, C_j 分别表示属于两个簇,但是有共同直接密度可达点的两个核心对象。 N_j, N_i 分别表示在对应核心对象的邻域内的对象的数目。 X_i, X_j 分别表示对应核心对象的邻域内的对象。

当 $RI(C_i, C_j) \leq 1$ 的时候,认为两个簇的互连性强,可以考虑合并成一个簇。当不满足这个条件的时候,考虑共享点的分配问题,采用距离密度函数的比来确定:

$$k = \frac{\sqrt{\sum (p - X_j)^2}}{N_j} \bigg/ \frac{\sqrt{\sum (p - X_i)^2}}{N_i}$$

其中: p 表示共享点, X_i, X_j 分别表示对应核心对象的

邻域内的对象, N_j, N_i 分别表示在对应核心对象的邻域内的对象的数目。当 $k < 1$ 时,将 p 划定到 C_i 对应的簇中;当 $k > 1$ 时,将 p 划定到 C_j 所对应的簇中。

5 算法分析

虽然算法增加了额外的运算。但是由于共享点远远小于数据点。它不会对时间复杂度有太大的影响。增加的运算时间可以忽略不计。为了防止可能出现的在密度均匀的时候可能出现的多个子簇的合并问题,经过实验发现当共享点的数量超过核心点的一半的时候聚类质量明显下降。所以,规定共享点不能超过核心点的一半。当超过的时候,近似按照原算法来进行处理,即谁先发现归谁所有。

6 实验结果说明及性能评估

实验结果说明及性能评估分别见表 1、图 1 和图 2。

表 1 两个算法的对比分析

数据量	MintPts	DBSCAN 算法得到的簇数	改进算法得到的簇数	改进的准确率
100	5	7	7	100 %
1000	5	12	10	92 %
10000	5	34	26	95 %
20000	5	21	14	90 %

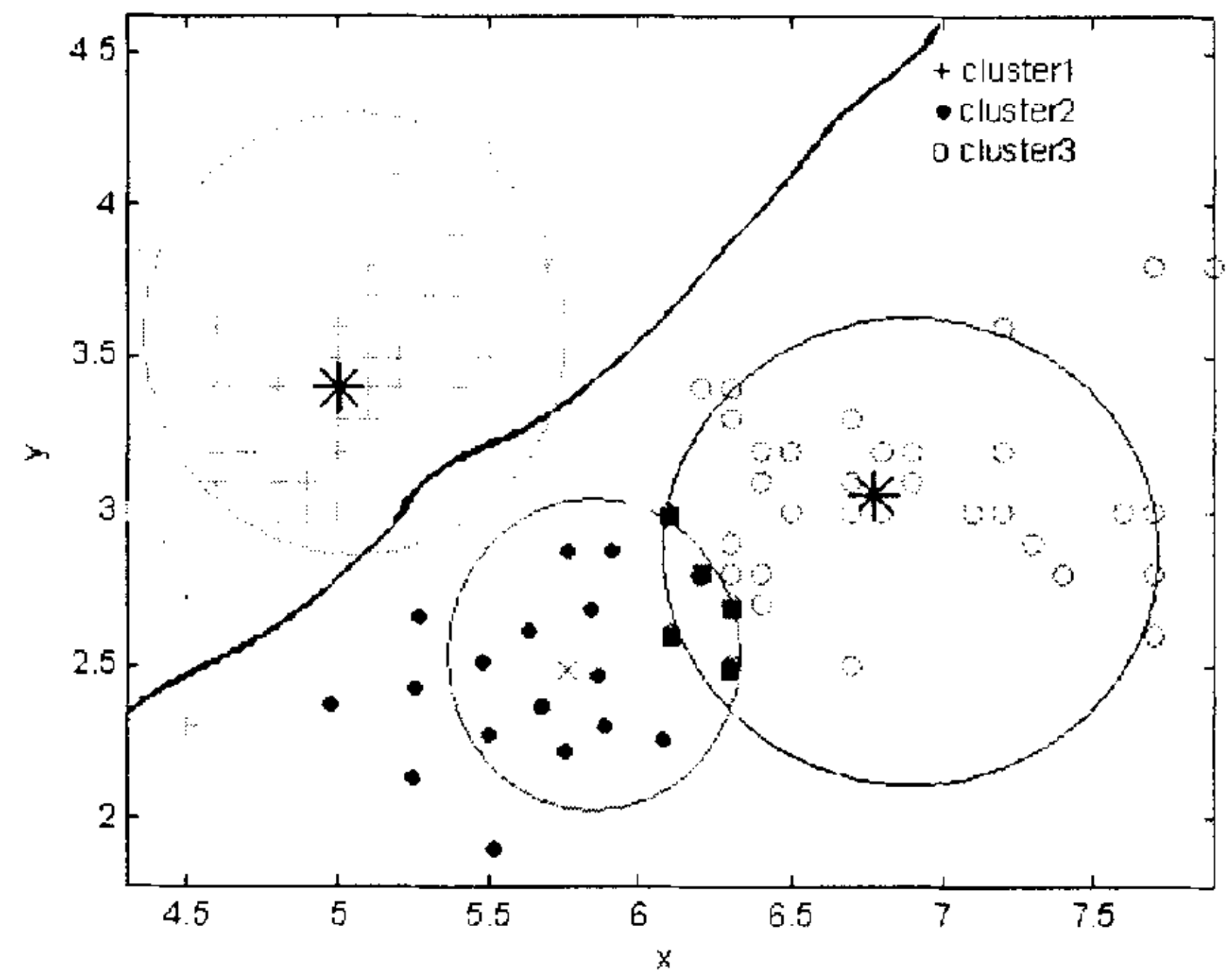


图 1 三个聚类中心的结果对比分析

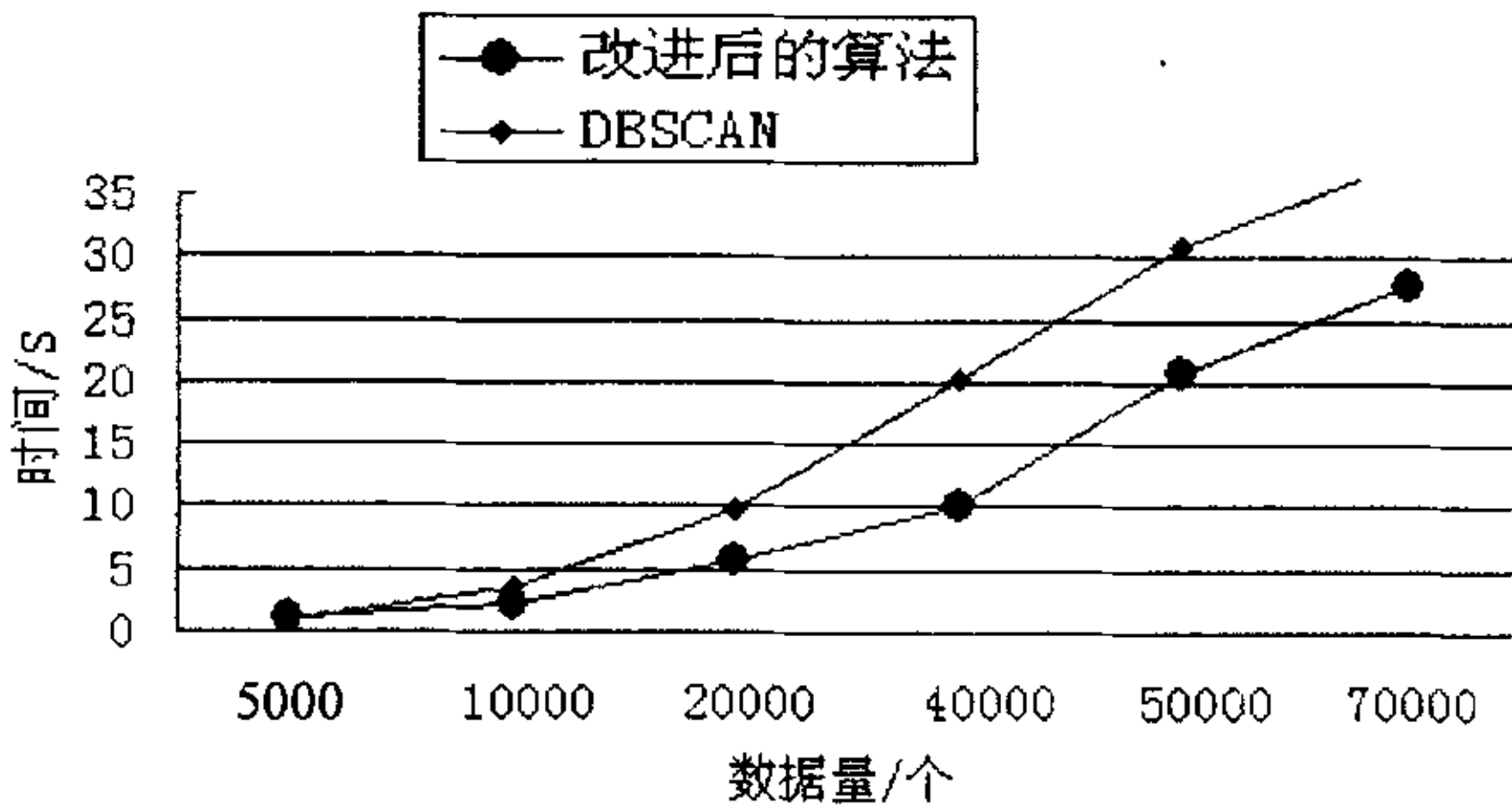


图 2 两种聚类方法的执行效率对比分析

时间。图 3(b) 是请求数量与服务正确响应率。图 4 是按照分解 1 得到的角色为 Role 本地流程 Process, 该虚拟服务组合包含调用风险评估 Web 服务, 类型为“分布式 1”的风险评估 Web 服务没有副本, 而“分布式 1*”有 1 个副本。

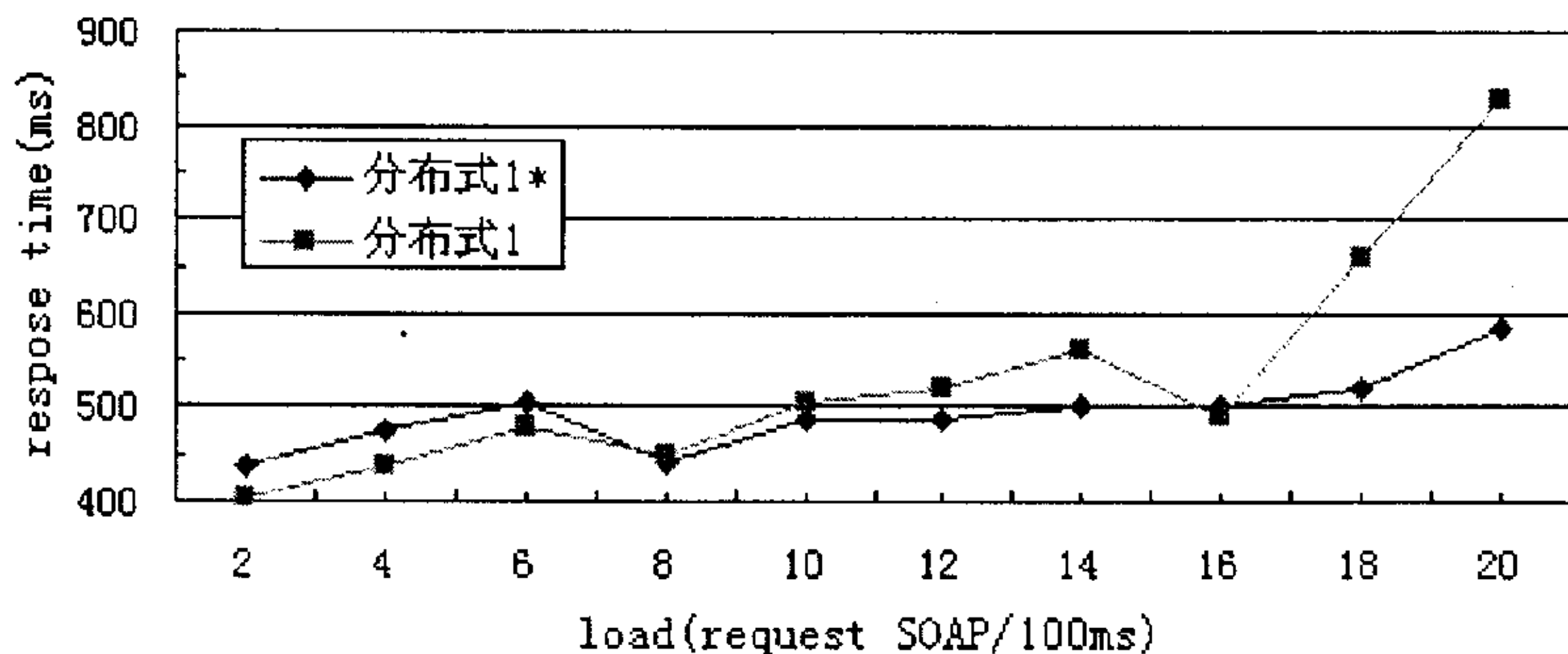


图 4 负载与响应时间在不同服务副本对比图。图 3 比较分布式与集中式服务组合, 在 10~16 (request SOAP/100ms) 中载和 16~18 (request SOAP/100ms) 重载时, 提供的服务组合优于集中式服务组合, 而分解 2 比分解 1 在重载时能提供 1100ms 响应时间和 90% 以上的正确执行的服务组合。图 4 测试了在随着服务副本增加使得在中载和重载时, 响应时间保持在 0.5s 左右。

4 结论和相关工作

应用分解算法分解产生 n (输入的角色数量) 个本地流程, 且本地流程没有绑定具体的 Web 服务, 如果假设, 某个本地流程接收客户 SOAP 请求, 该本地流程通过与其他 $n-1$ 个本地流程交互, 实现服务组合, 将结果返回给用户, 这种组合方式称为虚拟服务组合。因此在考虑分布式服务组合的 QOS 时, 应从 n 个本地流

程整体和局部方面考虑, 以及考虑负载均衡^[5], 并结合对客户的 SOAP 请求到达的概率, 考虑引擎在服务组合执行时的 QOS。现阶段在服务组合的 QOS 着重研究基于客户的 QOS, 依据此刻的服务组合和参与编排的 Web 服务的负载 (非该客户请求的服务组合运行时的负载), 根据算法 (如免疫算法^[6]) 进行服务副本选择。而负载均衡则实时均衡主机间的负载^[5]。在服务组合 QOS 的服务副本选择上考虑服务副本所在的主机的负载均衡问题。而这将成为服务组合的 QOS 研究的新热点。

参考文献:

- [1] 基于服务的建模和架构 [EB/OL]. 2004. <http://www-128.ibm.com/developerworks/cn/webservices/ws-soa-design1>.
- [2] Business Process Execution Language for Web Services. Version 1.1 [EB/OL]. 2003. <http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/bpel-ws.pdf>.
- [3] Nanda M G, Chandra S, Sarkar V. Decentralizing Execution of Composite Web Services [C] // Conference on Object Oriented Programming Systems Languages and Applications. New York: ACM Press, 2003: 170-187.
- [4] 刘必欣, 王玉峰, 贾 焰, 等. 一种基于角色的分布式动态服务组合方法 [J]. 软件学报, 2005, 16(10): 1859-1867.
- [5] 李文中, 郭 胜, 许 平, 等. 服务组合中一种自适应的负载均衡算法 [J]. 软件学报, 2006, 17(5): 1859-1867.
- [6] GAO Yan, NA Jun, ZHANG Bin, et al. Immune Algorithm for Selecting Optimum Services in Web Services Composition [J]. Wuhan University Journal of Natural Sciences, 2006, 11(1): 221-225.

(上接第 39 页)

通过实验结果可以看到采用了新的改进算法后可以将许多错误被分开的簇重新合并, 并可以保证合并的准确率在 90% 以上。事实证明此算法的改进是可行并有效的。同时由于采用的处理方法比较简单, 而且没有附加的磁盘 I/O 操作, 在算法的执行效率上比 OPTICS 要好得多。

7 总 结

基于密度的聚类算法 DBSCAN 是一种有效的算法, 由于它可以发现任意形状的簇而且能够较好地减少噪声的干扰, 因此能被广泛地应用。通过对此算法的改进, 弥补了其本身的不足, 因而可以更有效地利用此算法进行数据挖掘。

参考文献:

- [1] Han Jiawei, Kamber M. 数据挖掘概念与技术 [M]. 北京: 机械工业出版社, 2001.
- [2] Ankerest M, Breuning M M. OPTICS: ordering, points to identify the clustering structure [C] // Proc. ACM SIGMOD Int Conf on Management of Data. Philadelphia, PA: ACM Press, 1999: 49-60.
- [3] Jihong G, Shuigeng Z, Fuling B, et al. Scaling up the DBSCAN Algorithm for Clustering Large Spatial Databases Based on Sampling Technique [J]. Wuhan University Journal of Natural Sciences, 2001, 6(1-2): 467-473.
- [4] 蔡颖琨, 谢昆青, 马修军. 屏蔽了输入参数敏感性的 DBSCAN 改进算法 [J]. 北京大学学报, 2004, 40(3): 102-106.
- [5] 黄永平, 王丽珍. 考虑对象方向关系的密度聚类算法 [J]. 云南大学学报, 2004, 26(3): 216-219.